p129

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/864,109	05/24/2001	Gerald Francis McBrearty	AUS920000939US1	1454

35525 7590 12/17/2004

IBM CORP (YA)
C/O YEE & ASSOCIATES PC
P.O. BOX 802333
DALLAS, TX 75380

EXAMINER	
INGBERG, TODD D	

ART UNIT	PAPER NUMBER
2124	

DATE MAILED: 12/17/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/864,109	MCBREARTY ET AL.	
	Examiner	Art Unit	
	Todd Ingberg	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE ____ MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 August 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,2,4,5,7,8,10,12,13,15,16,18,20,21 and 23-32* is/are pending in the application.

4a) Of the above claim(s) *5,7,13,15,21 and 23* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,2,4,5,7,8,10,12,13,15,16,18,20,21 and 23-32* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *8/12/04* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Claims 1 – 2, 4 – 5, 7-8, 10, 12 – 13, 15 – 16, 18, 20 – 21, 23 - 32 have been examined.

Claims 1,3,6,9,11,14,17,19 and 22 have been amended.

Claims 5,7,13,15,21 and 23 have been canceled.

Claims 25 – 32 have been added

Drawings

1. Proposed amendment to Figures 1 and 2 has been accepted.

Well Known in the Art of Unix

2. The following should be old and well known prior to the time of invention if not part of the enabling technology for the invention.

A. **–g option in the C compiler** – performs analysis of the object code and symbol table as a result of the compilation process.

B. **DUMP** utility in UNIX.

Well Known in the Art of Debuggers

The following are considered to be well known in the art to one of ordinary skill in the art of Debuggers.

A. **The role of Symbol Tables in a Debugger.**
Loading a symbol table into a debugger is considered grossly old and well known. Prior art of record mentions the loading of the programs symbol table from the object file or the actual executable (Examiner is terming this internal since it comes from a byproduct of the compiler. Also the basis of the rejection is the loading of the symbol table externally by a routine that generates the symbol table is also well known. The invention is using the –g function of the C compiler. Examiner is calling this a form of external symbol table generation. Examiner presumes this is because a flat file is generated and easily used as input to generate the script. Also, well known is the loading of the symbol table in memory for use by the debugger.

B. **Contents of a Symbol Table (ST)**
One of ordinary skill in the art would understand the basic contents of a Symbol Table. The Examiner has made of record a portion of the September 12, 1985 book Compilers Principles, Techniques and Tools section on the Symbol Table pages 429 – 440.

Claim Rejections - 35 USC § 112

3. Rejection to claims 1 , 9 and 17 has been overcome by claim cancellation.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 1, 9, 17, 25 and 29 are rejected under 35 U.S. C. § 101for being directed to non-statutory subject matter. The Examiner has show one way to over come this rejection.

Claim 1
A method of debugging a software program executing on a computer and stored on **a computer readable medium** whose execution flow is responsive at least in part to a first set of options, said method comprising the steps of: generating a first log file by executing said program with said at least one of said first set of options, said first log file including an indication of all functions executed by said program during this first execution;
modifying at least one of the first set of options, and then generating a second log file by executing said program with said modified first set of options, said second log file including an indication of all functions executed by said program during this second execution; and comparing said first log file with said second log file to debug the software program.

Claim 9
A computer program product for debugging a software program **executing on a computer and stored on a computer readable medium** whose execution flow is responsive at least in part to a first set of options, said computer program product comprising: instruction means for generating a first log file by executing said program with said at least said first set of options, said first log file including an indication of all functions executed by said program during this first execution;
instruction means for modifying at least one of the first set of options, and then generating a second log file by executing said program with said first set of options, said second log file including an indication of all functions executed by said program during this second execution; and instruction means for comparing said first log file with said second log file to debug the software program.

Claim 17

A system for debugging a software program **executing on a computer and stored on a computer readable medium w**hose execution flow is responsive at least in part to a first set of options, comprising: a first log file being generated by executing said program with said first set of options, said first log file including an indication of all functions executed by said program during this first execution; a second log file being generated by executing said program with a modified first set of options, said second log file including an indication of all functions executed by said program during this second execution; and means for comparing said first log file with said second log file to debug the software program.

Claim 25
A method of debugging a software **program executing on a computer and stored on a computer readable medium,** said method comprising the steps of: analyzing said software program to determine a listing of executable functions in the software program; generating a file including said listing of executable functions; and generating a debug script for a debug program utilizing said file.

Claim 29
A computer program product **executing on a computer and stored on a computer readable medium** for debugging a software program, said computer program product comprising: instruction means for analyzing said software program to determine a listing of executable functions in the software program; instruction means for generating a file including said listing of executable functions; and instruction means for generating a debug script for a debug program utilizing said file.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1 – 2, 4 – 5, 7-8, 10, 12 – 13, 15 – 16, 18, 20 – 21, 23 - 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over The Borland Turbo Debugger building an external Symbol Table for use in a Debugger regardless of the environment (programming language or

Operating System (**OS**)) in view of building a debug test script for a debugger as taught in USPN # 6,161,216 Shagan.

It is deemed obvious to combine the teaching of building an external Symbol Table as taught by Borland with the teaching of building a Debugging Script from Shagan because the Symbol Table is a key source of information for determine what to test code (Shagan, col 2, coverage lines 45 – 46).

Note: Shagan uses the term "trace point" to represent a breakpoint.

Claim 1

A method of debugging a software program whose execution flow is responsive at least in part to a first set of options, said method comprising the steps of: generating a first log file by executing said program with said at least one of said first set of options, said first log file including an indication of all functions executed by said program during this first execution;

modifying at least one of the first set of options, and then generating a second log file by executing said program with said modified first set of options, said second log file including an indication of all functions executed by said program during this second execution; and comparing said first log file with said second log file to debug the software program.

Examiner's Rejection

Borland teaches the ability to build a Symbol Table externally is taught in the Borland Turbo Debugger. The file extension for the external symbol table is .TDS as on page 377 also see page 358 for the loading of the external table. The teaching of generating an external Symbol Table is on page 375. The Symbol Table in the debugger is loaded into memory Borland uses a Terminate and Stay Resident (TSR) routine (on first reading this might not be clear – be sure to read the memory mapping for symbols (pages 304 and 307) OR refer to prior art of record. It is the loading and comparison of the ST. The Borland reference provides evidence of this comparison when error messages are generated (page 378). In terms of options Borland teaches the ability to generate a Symbol Table (Option one – also includes external, import or no symbol table). In terms of the second option the option to run a debugger inherently uses a call graph which maps the possible execution paths. Shagam builds the "test script" (Shagam, Abstract) from the call graph. The Applicant has used the terms "log" for these well known structures. Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 2

The method according to claim 1, further comprising the steps of: generating said first log file including a first set of return codes; generating said second log file including a second set of return codes; and comparing said first set of return codes with said second set of return codes to debug the software program.
Examiner's Rejection

At this point in the claims the return codes are not specific enough a debug script. The intended use of return codes meets the current claim limitations as per Borland page 141.

Claim 3

The method according to claim 1, further comprising the step steps of. compiling said software program to generate compiled code, said compiled code including a listing of said functions, generating a file including said listing obtained from said compiled code; and generating a debug script utilizing said file.
Examiner's Rejection

The generation of a test script as per claim 1.

Claim 4

The method according to claim 3, further comprising the step of compiling said software program utilizing a C compiler and utilizing a "-g" option, said "-g" option generating said listing of said function.
Examiner's Rejection

Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 6

The method according to claim 3, further comprising the step of utilizing a UNIX dump command to generate said file, said UNIX dump command causing an output of said listing.
Examiner's Rejection

The building of external ST is taught by Borland and the use of the UNIX utility DUMP (Admitted prior art) is deemed obvious because the command lists information relevant to the programs functions. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art UNIX DUMP utility to generate information on Symbols. As currently claimed normal use of a well known UNIX utility. not clearly and concisely tied into actual invention.

Claim 8

The method according to claim 1, further comprising the steps of: automatically generating a debug script including the steps of: generating script code for each of a plurality of function calls included in said software program, said script code setting a breakpoint at each of said plurality of function calls; generating script code which logs each of a plurality of said plurality of

functions calls executed by said software program when said software program is executed under the control of said debug program; and generating script code which causes execution of said software program to continue after each of said plurality of said plurality of function calls is logged.

Examiner's Rejection

Shagan teaches placing breakpoints in the debug script (Col 1, Summary of invention) and specifically mentions where to place the break points including prior to calling a function (Col 2, lines 20 – 35).

Claim 9

A computer program product for debugging a software program whose execution flow is responsive at least in part to a first set of options, said computer program product comprising: instruction means for generating a first log file by executing said program with said at least said first set of options, said first log file including an indication of all functions executed by said program during this first execution;

instruction means for modifying at least one of the first set of options, and then generating a second log file by executing said program with said first set of options, said second log file including an indication of all functions executed by said program during this second execution; and instruction means for comparing said first log file with said second log file to debug the software program.

Examiner's Rejection

Borland teaches the ability to build a Symbol Table externally is taught in the Borland Turbo Debugger. The file extension for the external symbol table is .TDS as on page 377 also see page 358 for the loading of the external table. The teaching of generating an external Symbol Table is on page 375. The Symbol Table in the debugger is loaded into memory Borland uses a Terminate and Stay Resident (TSR) routine (on first reading this might not be clear – be sure to read the memory mapping for symbols (pages 304 and 307) OR refer to prior art of record. It is the loading and comparison of the ST. The Borland reference provides evidence of this comparison when error messages are generated (page 378). In terms of options Borland teaches the ability to generate a Symbol Table (Option one – also includes external, import or no symbol table). In terms of the second option the option to run a debugger inherently uses a call graph which maps the possible execution paths. Shagam builds the "test script" (Shagam, Abstract) from the call graph. The Applicant has used the terms "log" for these well known structures. Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 10

The product according to claim 9, further comprising: instruction means for generating said first log file including a first set of return codes; instruction means for generating said second log file including a second set of return codes; and instruction means for comparing said first set of return codes with said second set of return codes to debug the software program.

Examiner's Rejection

At this point in the claims the return codes are not specific enough a debug script. The intended use of return codes meets the current claim limitations as per Borland page 141.

Claim 11

The product according to claim 9, further comprising; instruction means for compiling said software program to generate compiled code, said compiled code including a listing of said functions instruction means for generating a file including said listing obtained from said compiled code; and instruction means for generating debug script utilizing said file.

Examiner's Rejection

The generation of a test script as per claim 9 and Shagan Figure 1, col 3, lines 25 – 50..

Claim 12

The product according to claim 11, further comprising instruction means for compiling said software program utilizing a C compiler and utilizing a "-g" option, said "-g" option generating said listing of said function.

Examiner's Rejection

Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 14

The product according to claim 11, further comprising instruction means for utilizing a UNIX dump command to generate said file, said UNIX dump command causing an output of said listing.

Examiner's Rejection

The building of external ST is taught by Borland and the use of the UNIX utility DUMP (Admitted prior art) is deemed obvious because the command lists information relevant to the programs functions. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art UNIX DUMP utility to generate information on Symbols. As currently claimed normal use of a well known UNIX utility. not clearly and concisely tied into actual invention.

Claim 16

The product according to claim 9, further comprising: instruction means for automatically generating a debug script including: instruction means for generating script code for each of a plurality of function calls included in said software program, said script code setting a breakpoint at each of said plurality of function calls; instruction means for generating script code which logs each of a plurality of said plurality of functions calls executed by said software program when said software program is executed under the control of said debug program; and instruction means for generating script code which causes execution of said software program to continue after each of said plurality of said plurality of function calls is logged.

Examiner's Rejection

Shagan teaches placing breakpoints in the debug script (Col 1, Summary of invention) and specifically mentions where to place the break points including prior to calling a function (Col 2, lines 20 – 35).

Claim 17

A system for debugging a software program whose execution flow is responsive at least in part to a first set of options, comprising: a first log file being generated by executing said program with said first set of options, said first log file including an indication of all functions executed by said program during this first execution; a second log file being generated by executing said program with a modified first set of options, said second log file including an indication of all functions executed by said program during this second execution; and means for comparing said first log file with said second log file to debug the software program.

Examiner's Rejection

Borland teaches the ability to build a Symbol Table externally is taught in the Borland Turbo Debugger. The file extension for the external symbol table is .TDS as on page 377 also see page 358 for the loading of the external table. The teaching of generating an external Symbol Table is on page 375. The Symbol Table in the debugger is loaded into memory Borland uses a Terminate and Stay Resident (TSR) routine (on first reading this might not be clear – be sure to read the memory mapping for symbols (pages 304 and 307) OR refer to prior art of record. It is the loading and comparison of the ST. The Borland reference provides evidence of this comparison when error messages are generated (page 378). In terms of options Borland teaches the ability to generate a Symbol Table (Option one – also includes external, import or no symbol table). In terms of the second option the option to run a debugger inherently uses a call graph which maps the possible execution paths. Shagam builds the "test script" (Shagam, Abstract) from the call graph. The Applicant has used the terms "log" for these well known structures. Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 18

The system according to claim 17, further comprising: said first log file being generated including a first set of return codes; said second log file being generated including a second set of return codes; and means for comparing said first set of return codes with said second set of return codes to debug the software program.

Examiner's Rejection

At this point in the claims the return codes are not specific enough a debug script. The intended use of return codes meets the current claim limitations as per Borland page 141.

Claim 19

The system according to claim 17, further comprising said software program being compiled to generate compiled code, said compiled code including a listing of said functions; a file being

generated including said listing obtained from said compiled code; and a debug script being generated utilizing said file.
Examiner's Rejection
The generation of a test script as per claim 9 and Shagan Figure 1, col 3, lines 25 – 50..

Claim 20
The system according to claim 19, further comprising said software program being compiled utilizing a C compiler and utilizing a "-g" option, said "-g" option generating said listing of said function.
Examiner's Rejection
Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 22
The system according to claim 24 9 further comprising a UNIX dump command being utilized to generate said file, said UNIX dump command causing an output of said listing.
Examiner's Rejection
The building of external ST is taught by Borland and the use of the UNIX utility DUMP (Admitted prior art) is deemed obvious because the command lists information relevant to the programs functions. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art UNIX DUMP utility to generate information on Symbols. As currently claimed normal use of a well known UNIX utility. not clearly and concisely tied into actual invention.

Claim 24
The system according to claim 17, further comprising: a debug script being automatically generated including: script code being generated for each of a plurality of function calls included in said software program, said script code setting a breakpoint at each of said plurality of function calls; script code being generated which logs each of a plurality of said plurality of functions calls executed by said software program when said software program is executed under the control of said debug program; and script code being generated which causes execution of said software program to continue after each of said plurality of said plurality of function calls is logged.
Examiner's Rejection
Shagan teaches placing breakpoints in the debug script (Col 1, Summary of invention) and specifically mentions where to place the break points including prior to calling a function (Col 2, lines 20 – 35).

Claim 25
A method of debugging a software program, said method comprising the steps of: analyzing said software program to determine a listing of executable functions in the software program;

generating a file including said listing of executable functions; and generating a debug script for a debug program utilizing said file.

Examiner's Rejection

Borland teaches the ability to build a Symbol Table externally is taught in the Borland Turbo Debugger. The file extension for the external symbol table is .TDS as on page 377 also see page 358 for the loading of the external table. The teaching of generating an external Symbol Table is on page 375. The Symbol Table in the debugger is loaded into memory Borland uses a Terminate and Stay Resident (TSR) routine (on first reading this might not be clear – be sure to read the memory mapping for symbols (pages 304 and 307) OR refer to prior art of record. It is the loading and comparison of the ST. The Borland reference provides evidence of this comparison when error messages are generated (page 378). In terms of options Borland teaches the ability to generate a Symbol Table (Option one – also includes external, import or no symbol table). In terms of the second option the option to run a debugger inherently uses a call graph which maps the possible execution paths. Shagam builds the "test script" (Shagam, Abstract) from the call graph. The Applicant has used the terms "log" for these well known structures. Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 26

The method of Claim 25, wherein the step of generating a debug script comprises: utilizing said file, including said listing of executable functions, to generate script code which logs executable functions executed by said software program when said software program is executed under the control of said debug program.

Examiner's Rejection

The generation of a test script as per claim 25.

Claim 27

The method of Claim 26, further comprising: setting at least one parameter used by the software program; executing said software program using said debug script to generate a first log file; changing the at least parameter and then re-executing said software program using said debug script to generate a second log file; and comparing said first log file and said second log file to debug the software program.

Examiner's Rejection

Shagam generating a "test script" (Shagam, Abstract). "Re-executing" - Interpreted as normal use of running a debugger with start and stop and the ability to start again. (Borland pages 97 and 98).

Claim 28

The method according to Claim 25, wherein the analyzing step is done by a compiler, and further comprising the step of utilizing a dump command to generate said file, said dump command causing an output of said listing.

Examiner's Rejection
The building of external ST is taught by Borland and the use of the UNIX utility DUMP (Admitted prior art) is deemed obvious because the command lists information relevant to the programs functions. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art UNIX DUMP utility to generate information on Symbols. As currently claimed normal use of a well known UNIX utility. not clearly and concisely tied into actual invention.

Claim 29
A computer program product for debugging a software program, said computer program product comprising: instruction means for analyzing said software program to determine a listing of executable functions in the software program; instruction means for generating a file including said listing of executable functions; and instruction means for generating a debug script for a debug program utilizing said file.
Examiner's Rejection
Borland teaches the ability to build a Symbol Table externally is taught in the Borland Turbo Debugger. The file extension for the external symbol table is .TDS as on page 377 also see page 358 for the loading of the external table. The teaching of generating an external Symbol Table is on page 375. The Symbol Table in the debugger is loaded into memory Borland uses a Terminate and Stay Resident (TSR) routine (on first reading this might not be clear – be sure to read the memory mapping for symbols (pages 304 and 307) OR refer to prior art of record. It is the loading and comparison of the ST. The Borland reference provides evidence of this comparison when error messages are generated (page 378). In terms of options Borland teaches the ability to generate a Symbol Table (Option one – also includes external, import or no symbol table). In terms of the second option the option to run a debugger inherently uses a call graph which maps the possible execution paths. Shagam builds the "test script" (Shagam, Abstract) from the call graph. The Applicant has used the terms "log" for these well known structures. Borland teaches a technique for building external Symbol Tables. External Symbol Tables are taught in the prior art and are deemed obvious regardless of the utility to build them because ST enable debuggers to identify the symbols in a program. Therefore, it would have been obvious to one of ordinary skill in the art at the time of invention to use the admitted prior art C compiler option to generate an external ST.

Claim 30
The computer program product of Claim 29, wherein the instruction means for generating a debug script comprises: instruction means for utilizing said file, including said listing of executable functions, to generate script code which logs executable functions executed by said software program when said software program is executed under the control of said debug program.
Examiner's Rejection
The generation of a test script as per claim 29.

Claim 31

The computer program product of Claim 30, further comprising: instruction means for setting at least one parameter used by the software program; instruction means for executing said software program using said debug script to generate a first log file; instruction means for changing the at least parameter and then re-executing said software program using said debug script to generate a second log file; and instruction means for comparing said first log file and said second log file to debug the software program.
Examiner's Rejection
Shagam generating a "test script" (Shagam, Abstract). "Re-executing" - Interpreted as normal use of running a debugger with start and stop and the ability to start again. (Borland pages 97 and 98).

Claim 32
The computer program product according to Claim 29, wherein the instruction means for analyzing said software program is a compiler, and further comprising a dump command means for generating said file, said dump command means causing an output of said listing.
Examiner's Rejection
The generation of a test script as per claim 29. Dump being the use of the cal graph to make the test script.

Conclusion

7. Applicant's Representative called the Examiner several months ago to schedule a Telephonic Interview. The Examiner had to discourage the Interview because problems receiving FAXES. Due to this customer unfriendly situation which has been corrected with the move to the new PTO campus this action is non final.

Correspondence Information

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Todd Ingberg** whose telephone number is **(571) 272-3723** (as of October 23, 2004).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, **Kakali Chaki** can be reached on **(571) 272-3719**. Please, note that as of August 4, 2003 the **Official FAX number** changed to **(703) 872-9306**.

Also, be advised the United States Patent Office **new address** is

Post Office Box 1450

Alexandria, Virginia 22313-1450

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Todd Ingberg
Primary Examiner
Art Unit 2124
December 12, 2004